CLAIMS:

We claim:

1. An event notification and management system comprising:

an aggregation of logical components, each logical component having a coupling to a corresponding dynamic proxy;

an event notification service communicatively linked to a plurality of subscribing processes; and,

an event queue disposed between said dynamic proxies and said event notification service.

2. The system of claim 1, wherein at least one of said dynamic proxies comprises at least one of event notification logic and event management logic.

3. The system of claim 1, further comprising an event list coupled to at least one of said dynamic proxies.

4. The system of claim 1, further comprising an event-to-subscriber list coupled to said event notification service.

5. The system of claim 1, further comprising an event action list coupled to at least one of said dynamic proxies.

6. A dynamic proxy configured for interoperation with a component instance in a dynamic aggregation of components, the dynamic proxy comprising:

a list of selected listener method calls in said component instance;

a communicatively coupling to an event queue; and,

event notification logic coupled to said list and configured to post events to said event queue which relate to invoked listener method calls included in said list.

7. The dynamic proxy of claim 6, further comprising event management logic configured to selectively handle invoked listener method calls.

8. The dynamic proxy of claim 7, wherein said event management logic comprises programming for selectively performing one of quashing an invoked one of said listener method calls, handling said invoked one of said listener method calls without passing said invoked one of said listener method calls to the component instance, assisting the component instance in handling said invoked one of said listener method calls while passing said invoked one of said listener method calls to the component instance, and modifying said invoked one of said listener method calls before passing said invoked one of said listener method calls to the component instance.

9. An event notification and management method comprising the steps of:

creating a component instance from an amalgamation of a dynamic proxy object definition and a component interface;

trapping selected calls to methods disposed within said component instance;

routing a reference to said trapped selected calls to an event queue; and,

for each reference in said queue, distributing a notification to a set of subscribers registered to receive notifications for said reference.

10. The method of claim 9, wherein said creating step comprises the step of instructing a factory object coupled to said dynamic proxy object definition to create said component instance.

11. The method of claim 9, further comprising the steps of:

determining whether an event notification service responsible for said distributing step has been activated; and,

performing said routing step only if said event notification service has been activated.

12. The method of claim 9, wherein said routing step comprises the steps of:

consulting an event list enumerating specific ones of said calls; and,

posting to said event queue only specific ones of said calls included in said event list.

13. The method of claim 9, further comprising the step of selectively performing one of quashing said trapped selected calls, handling said trapped selected calls without passing said trapped selected calls to said component instance, assisting said component instance in handling said trapped selected calls while passing said trapped

selected calls to said component instance, and modifying said trapped selected calls before passing said trapped selected calls to said component instance.

14. A machine readable storage having stored thereon a computer program for event notification and management, the computer program comprising a routine set of instructions which when executed by a machine cause the machine to perform the steps of:

creating a component instance from an amalgamation of a dynamic proxy object definition and a component interface;

trapping selected calls to methods disposed within said component instance;

routing a reference to said trapped selected calls to an event queue; and,

for each reference in said queue, distributing a notification to a set of subscribers registered to receive notifications for said reference.

15. The machine readable storage of claim 14, wherein said creating step comprises the step of instructing a factory object coupled to said dynamic proxy object definition to create said component instance.

16. The machine readable storage of claim 14, further comprising the steps of:

determining whether an event notification service responsible for said distributing step has been activated; and,

performing said routing step only if said event notification service has been activated.

17. The machine readable storage of claim 14, wherein said routing step comprises the steps of:

consulting an event list enumerating specific ones of said calls; and,

posting to said event queue only specific ones of said calls included in said event list.

18. The machine readable storage of claim 14, further comprising the step of selectively performing one of quashing said trapped selected calls, handling said trapped selected calls without passing said trapped selected calls to said component instance, assisting said component instance in handling said trapped selected calls while passing said trapped selected calls to said component instance, and modifying said trapped selected calls before passing said trapped selected calls to said component instance.